Introduction to Programming

Revision
2019 - 2020

About the Exam

- Online, weighs 70% (20% In-lab test, 10% attendance)
- Date and Time: Tuesday 26th May 2020, 02.00pm – 06.00pm (Moodle time)
  - 30 min for downloading the questions
  - 3 hours for completing the answers
  - 30 min for uploading the answers
  - 4 hours in total (5 hours in total if you have an SSP*)
  - The submission dropbox will close after the deadlines

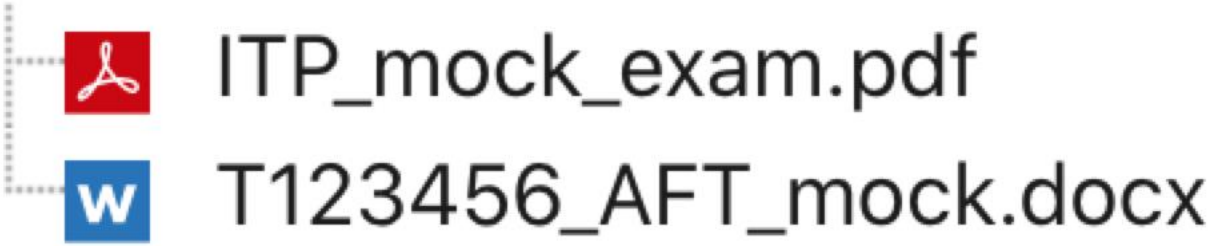
* SSP: Study Support Plan

About the Exam

- Open book
  - but no copying, no collusion
- 10 questions, 100 marks
- Memorize, Explain, Compare, Analyse, Work out, Program

About the Exam

- What's available at the start of the exam?
  - An examination question paper in pdf format
  - An answer sheet template in Word format

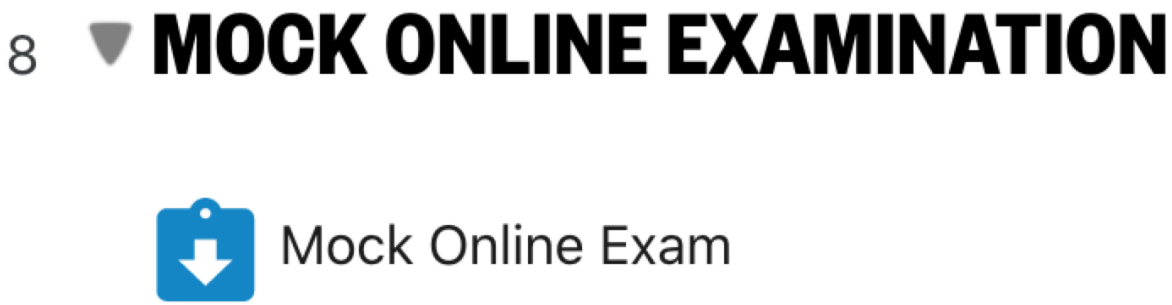


About the Exam

- What's to submit at the end of the exam?

- One PDF file (≤ 100MB) containing ALL the answers

- Moodle Assignment submission portal on the course's Moodle page



About the Answer Sheet

- Word or other text editors would be sufficient

  (e.g. Libre Office on Linux, Google Docs, TextEdit on Mac)
  - No need to draw graphs or tables
- In the answer sheets
  - Use blue colour (avoid using black colour)
  - Become familiar with operations such as indentation, text formatting, and the insertion of images, diagrams or pictures (if needed), etc.
  - No need to format your answers perfectly -> content over form
- No matter which text editor you use, save as one PDF file
  - Pdf files are more reliable and less likely to be changed
  - One file to prevent answers being overlooked or missing

All Hand-Written Notes

- Hand-written notes (only choose to do this if there are no other options)
  - Option 1:
    - Take pictures of your handwritten answers
    - Open a Word document and insert your pictures
    - Save as a PDF file and upload it to Moodle
  - Option 2:
    - (Download a free App on your smart phone: CamScanner)
      - You may use other Apps or a scanner
    - Scan all your answer sheets one by one into one document
    - Save as PDF and email it to yourself and Upload it to Moodle

Name the Answer Sheet

- Naming your answer sheet
  - Anonymous marking, as always
  - Do NOT put any personal information in the title or in the file
  - Use your candidate number, e.g., T123456 (can be found in student profile)
    - candidateNumber_AFT.pdf or candidateNumber_APT.pdf or candidateNumber_BPT.pdf
    - Example: T123456_AFT.pdf or T123456_APT.pdf or T123456_BPT.pdf
- A mock exam is available to practice the whole process

Hardware and Environment

- Hardware and network
  - Adequate desktop or laptop
  - A reliable broadband connection is required
  - Consider a backup internet connection
    - Mi-Fi device/dongle (aka, Pocket WiFi) or a tether to a mobile phone
- Choose a comfortable and quiet room
  - Print out the questions if it helps

In Case Things Go Wrong

- The Department will inform you about alternative arrangements in case uploading to Moodle fails.
- For technical queries specifically relating to the following issues please contact ITS.
  - Logging into Moodle
  - Scanning handwritten work
  - Submitting assessments
  - Email: sd@its.bbk.ac.uk
  - Phone: 020 3926 3456
- Save a copy of the pdf examination file at your local disc. Do NOT change the file. This can be the evidence if things go wrong.

Prepare for the Exam

- Study the lecture slides
  - Refer to the book PFE if needed

- Study the past papers
  - Similar structure and marking scheme
  - Available online at Birkbeck's electronic library
    - http://www.bbk.ac.uk/library/exam-papers/computer-science
  - Some summary answers available on the course website
    - https://www.dcs.bbk.ac.uk/~sjmaybank/ITP/introduction%20to%20Programming.html

Week 1: First Program

- Basic knowledge on Python
  - History, advantages, interpreter, portability
- Errors
  - Compile time errors, run time exceptions/errors
    - What are they? Examples?
- Function print
  - print(a number) √ — print(5)
  - print(a string) √ — print("hello")
  - print(string, number) √ — print("hello", 5)
  - print(string + number) X — print("hello" + 5) X

Week 2b: Variables

Birkbeck UNIVERSITY OF LONDON

- Variables
  - Creation and value assignment
    - A variable can never be used if not created and initialised
  - Identify a variable's name, trace a variable's value
  - Variable naming rules
    - begin with? the rest? reserved words?
  - Creating variables for problem solving

- Number Literals
  - int: 1, 0, -2, etc
  - float: 2.0, 8E4, 3e-5, etc

Week 3: Arithmetic and Built-in Functions

Birkbeck UNIVERSITY OF LONDON

- Operators
  - \+ – * ** / // % ( )
  - Precedence
    - ( ) higher than ** higher than *, /, //, % higher than +, –
  - Associativity
    - **: right to left (e.g., p = 2 ** 2 ** 3)
    - Other operators: left to right
- Built-in functions
  - abs, min, max, round (round up/down at half point)
  - Nested (built-in) functions, e.g., round(max(num_1,num_2))
- Dividing a problem into a sequence of simple steps

Week 4: More Arithmetic and Input (2)

Birkbeck UNIVERSITY OF LONDON

- Evaluate expressions
  - e.g., a = 1, b = 2, a = b – a * b, what is a?
  - b = b – (a + 3) * 4, what is b?
- Math module
  - common math functions, sqrt, exp, trunc, etc
  - How to obtain a math function?
    - from math import * (red indicates reserved words)
    - A math function cannot be used if the import is not called (as above).

Week 4: More Arithmetic and Input (2)

Birkbeck UNIVERSITY OF LONDON

- User input
  - userInput = input("Please enter a number: ")
  - userInput is a string
  - How to turn the string into int or float?
    - function int() and float()
    - print(int("5.6")) error, but print(float("5")) works
    - print(userInput)
- Round-off errors
  - Why some numbers cannot be represented exactly in Python?
  - E.g., 4.35 * 100 != 435
- Write programs to solve detailed problems

Week 5: Strings and Output (1)

- Strings
  - length, indexing (positive/negative)
  - concatenation (+), repetition (*)
  - string and print
  - convert numbers to strings str(num)
    - print(5) or print(5.0) ok
    - print(len("hello") + 5) 10
    - print("hello"+ str(5)) ok
    - print("hello"+ str(5.0)) ok
  - escape sequences \", \', \n, \\
    - each with length 1

Week 5: Strings and Output (2)

- Strings

```
print('He\\ said "Hello" today')
```

The double quotes " are characters in the string

Result: He\ said "Hello" today

```
print("He said 'Hello' today\n")
```

The single quotes ' are characters in the string

Result: He said 'Hello' today

[a new line]

```
print("He said \"Hello\" and 'Goodbye' today")
```

The single quotes ' are characters in the string

Result: He said "Hello" and 'Goodbye' today

What about the length of the above strings?

Week 5: Strings and Output (3)

- Format specifiers
  - Be able to identify
    - a format specifier, a format string and a string format operator
    - "%.f" % 35.678
      - format specifier: %.f
      - format string: "%.f"
      - string format operator: %
  - Be able to apply a format string to a value
    - formatString % value
    - e.g., print("%.f" % 35.678)
  - Understand how a format specifier works
    - See next slide

Format Specifier Summary

formatString % value

~ represents a space

formatString		value		
		(float) 35.678	(integer) -5	(string) "hello"
float (round)	"%.f"	"36"	"-5"	error
	"%.2f"	"35.68"	"-5.00"	
	"%6.1f"	"~~35.7"	"~~-5.0"	
	"%07.2f"	"0035.68"	"-005.00"	
integer (trunc)	"%d"	"35"	"-5"	error
	"%5d"	"~~~35"	"~~~-5"	
string	"%s"	"35.678"	"-5"	"hello"
	"%7s"	"~35.678"	"~~~~~-5"	"~~hello"
	"%3s"	"35.678"	"~-5"	"hello"

e.g., print("%.f" % 35.678), print("%d" % -5), or print("%s" % "hello")

Format Specifier Practice Sheet

formatString % value ~ represents a space

formatString		value		
		(float) 35.678	(integer) -5	(string) "hello"
float (round)	"%.f"			
	"%.2f"			
	"%6.1f"			
	"%07.2f"			
integer (trunc)	"%d"			
	"%5d"			
string	"%s"			
	"%7s"			
	"%3s"			

e.g., print("%.f" % 35.678), print("%d" % -5), or print("%s" % "hello")

Week 6: Relational Operators and Boolean Variables

- Relational operators
  - >, >=, <, <=, ==, !=
- Boolean variables
  - Boolean values: True, False
  - Boolean operators: and, or, not
    - truth table (what it is, how to write a truth table)
- Evaluate a Boolean expression
  - 73 == 9, 73 <= 9, etc

Week 6: Relational Operators and Boolean Variables (2)

Birkbeck UNIVERSITY OF LONDON

- Define a Boolean expression
  - E.g., an expression is true if and only if all the variables a, b, c are true
  - a and b and c

  - E.g., an expression is true if and only if the variables b is false or a is true
  - not b or a
  - (not b) or a

  - E.g., an expression is true if and only if the variables b is false
  - not b

Week 6: Relational Operators and Boolean Variables (3)

- Lexicographic ordering of characters
  - How is the order of characters defined in python?
    - uppercase < lowercase
    - numbers < letters
    - space < printable
    - empty string < non-empty characters

"" < " " < "0" < "1" < "9" < "A" < "B" < "Z" < "a" < "b" < "z"

- Lexicographic ordering of strings
  - How are strings compared in Python?
  - "cart" < "car" - True or False?

Week 7: if Statement

Birkbeck UNIVERSITY OF LONDON

- Learn and apply the following statements:
  - if
  - if-else
  - nested if-else
  - if-elif-else
- Indentation plays an important role
- Input validation + error message
- Never forget the :

Week 8: Loops

Birkbeck UNIVERSITY OF LONDON

- range() function and its use in for-loops
  - range(100), range(2,9), range(2, 9, 2)
  - i in range(100)
- while-loop and for-loop
  - When to use while-loop, when to use for-loop
  - How to rewrite while-loop to for-loop and vice versa
  - Use while-loops to control how many times it loops
- Use while-loops and for-loops to solve problems

For loop to while loop

```
sum = 0
for index in range(0, 101, 2):
        sum = sum + index
print(sum)
```

```
sum = 0
index = 0
while index <= 100 :
        sum = sum + index
        index = index + 2
print(sum)
```

Week 9: Functions

Birkbeck UNIVERSITY OF LONDON

- Functions
  - What are function *name*, *parameter*, *argument*, *return*?
  - What is the header/body of a function?
  - Why to define a function? The advantage of using functions
  - Defining functions and calling functions
  - Branches and returns
  - Local variables and scope of a variable
  - Write functions to solve problems

Week 11: Lists

Birkbeck UNIVERSITY OF LONDON

- What is a list?
- Why need lists?
- How to create a list?
- List indices and lengths
  - values[3]? values[4]? values[-3]? values[-4]?
  - len(values)? len(names)?
- Finding/Dealing with elements in lists
  - index – names.index("Ben") the index of 1st occurrence
  - append – values.append(89) or names.append("Dylan")
  - insert – values.insert(1, 2) or names.insert(2, "Finn")
  - remove – values.pop(3) or names.pop()
  - What is names[1][1]?

values = [32, 54, 67, 5]

names=["Ann", "Ben", "Chris"]

Last but not least…

- If you see "Justify your answer" or "Specify the reason", do provide some explanations, otherwise, mark(s) will be deducted.
- If you see "Write down the step-by-step results of all calculation", do provide all the intermediate results.
- Read questions carefully. Don’t be in a rush.
- Think carefully.
- Write clearly and to the point.
  - Please don’t write essays! ☺
- Good luck!